# Empirical Analysis of Support Vector Machine Ensemble Classifiers

Shi-jin Wang[a]*, Avin Mathew[b], Yan Chen[c], Li-feng Xi[a], Lin Ma[b], Jay Lee[c]

[a] *Department of Industrial Engineering & Management, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China*
[b] *Cooperative Research Centre for Integrated Engineering Asset Management (CIEAM), Queensland University of Technology, Brisbane, Australia*
[c] *NSF Center for Intelligent Maintenance Systems (IMS), University of Cincinnati, Cincinnati, U.S.A*

## Abstract

Ensemble classification – combining the results of a set of base learners – has received much attention in the machine learning community and has demonstrated promising capabilities in improving classification accuracy. Compared with neural network or decision tree ensembles, there is no comprehensive empirical research in support vector machine (SVM) ensembles. To fill this void, this paper analyses and compares SVM ensembles with four different ensemble constructing techniques, namely bagging, AdaBoost, Arc-X4 and a modified AdaBoost. Twenty real-world data sets from the UCI repository are used as benchmarks to evaluate and compare the performance of these SVM ensemble classifiers in terms of the classification accuracy. Different kernel functions and different numbers of base SVM learners are tested in the ensembles. The experimental results show that although SVM ensembles are not always better than a single SVM, the SVM bagged ensemble performs as well or better than other methods with a relatively higher generality, particularly SVMs with a polynomial kernel function. Finally, an industrial case study of gear defect detection is conducted to validate the empirical analysis results.

## Keywords

Ensemble classification; Support vector machines (SVMs); AdaBoost; Bagging; Classification

## 1. Introduction

The pursuit of higher accuracy has been a driving force in directing research into machine learning. Ensemble classification learning generates a set of base classifiers (or inducer algorithms) using different distributions of training data and then aggregates their outputs to classify new samples. These ensemble learning methods enable users to achieve more accurate predictions with higher generalization abilities than the predictions generated by individual models or experts on average (Wezel and Potharst, 2007).

There are two popular approaches for constructing ensemble classifiers: bagging (Breiman, 1996) and boosting (Freund, 1995). The majority of existing theoretical and empirical research have investigated the underlying mechanism of bagging or its variants (e.g., random forest (Breiman, 2001)), and boosting or its variants (e.g. AdaBoost (Freund and Schapire, 1997; Freund and Schapire, 1999)).

The generalization performance of ensemble classifiers is dependent on the diversity and accuracy trade-off of the base classifiers. Both bagging and boosting realize this trade-off by minimizing the classification error on different parts of the input space via intrinsic "resampling" technique. The main difference between them is that boosting adaptively changes the distribution of the training set based on the performance of previous classifiers while bagging does not (Bauer and Kohavi, 1999). From the aspect of the bias-variance decomposition of the error rate, some researchers believe that AdaBoost can outperform bagging in both bias and variance errors (Webb and Zheng, 2004), while others conclude

* Corresponding author. Tel: +86-021-54748366; Fax: +86-021-34206539;
Email: shijinwang0223@yahoo.com.cn

that AdaBoost is more effective at reducing bias than bagging and that bagging is more effective at reducing variance (Bauer and Kohavi, 1999; Webb, 2000). However, there is still no single account that has received undisputed widespread support (Webb and Zheng, 2004).

The theoretical research into the ensemble techniques mentioned above was mostly deduced through empirical analysis. For example, Opitz and Maclin (1999) used both neural networks and decision trees as base classifiers to study the performance of bagging and boosting (Arc-x4 and AdaBoost), through 23 data sets with different numbers of ensembles. Their empirical analysis results indicate that bagging is almost always more accurate than a single classifier, although it can be considerably less accurate than boosting. They also noted that the performance of boosting methods is dependent on the characteristics of the data set, and discovered that most of the gain in an ensemble's performance comes in the first few classifiers combined. Bauer and Kohavi (1999) conducted a performance comparison of bagging, bagging variants, AdaBoost and Arc-x4 on decision tree and naive Bayes classifiers, using 14 large-scale data sets from UCI. Based on experimental results, they found that the boosting algorithms were generally better than bagging, but not uniformly better. They also found that for some data sets, ensembles did not improve the classification performance. Schwenk and Bengio (2000) investigated neural network ensembles using AdaBoost and its three variants. The experimental results of three real-world applications demonstrated the effectiveness of their AdaBoost ensemble. Based on the experimental analysis, they also discovered the sensitivity of AdaBoost to overtraining of individual classifiers. Banfield et al. (2007) experimentally evaluated bagging and other seven randomization-based approaches (including boosting, random subspaces, randomized C4.5 and random forests) of decision tree ensembles with a large number of data sets using 10-fold cross validation and $5 \times 2$-fold cross validation. Based on the experimental results, they found that the best method was statistically more accurate than bagging on only 8 of the 57 data sets and that boosting, random forests and randomized trees were statistically more accurate than bagging on average.

Most existing empirical analysis of ensembles mentioned above use weak learners (e.g. decision trees, neural networks, or naive Bayes) in PAC (probably approximately correct) learning theory. As the typical goal of learning classification methods is to maximize classification accuracy with a higher generalization ability, it is important to examine ensembles based on non-weak classifiers, such as support vector machines (SVM).

Support vector machines are a new generation learning system based on recent advances in statistical learning theory (Cristianini and Shawe-Taylor, 2000). SVMs calculate a separating hyperplane that maximizes the margin between data classes to produce good generalization abilities. SVMs have proved to be an efficient learning machine from numerous successful applications (Hsu and Lin, 2002; Widodo et al., 2007; Widodo and Yang, 2007). However, despite its high performance, SVMs have some limitations. For example, the performance of multi-class classification cannot match that of binary classification as SVMs use approximation algorithms to reduce the computation complexity but these have the effect of degrading classification performance (Kim et al., 2003). Consequently, researchers have attempted to further enhance SVMs with ensemble techniques. Valentini and Dietterich (2004) showed that bias-variance decomposition offers a rationale to develop SVM ensembles, and they proposed two directions for developing SVM ensembles: bagged ensembles of selected low-bias SVMs and heterogeneous ensembles of SVMs. In their subsequent research, they evaluated and quantitatively measured the bias-variance decomposition of error in ensembled SVMs (Valentini, 2005). Pang et al. (2003) indicated that SVM ensembles are a type of cross-validation optimization of single SVM, and should have a more stable classification performance than other models. Their research involved using SVM ensembles in membership authentication. To improve the limited classification performance of SVMs, Kim et al. (2003) used bagging, AdaBoost and three aggregation methods (majority vote, LSE-based weighting and double-layer hierarchical combining) to construct SVM ensembles. The resulting SVM ensembles fared

better than a single SVM when tested against two UCI data sets and a data set on cellular fraud in the telecommunications industry. Li et al. (2005) examined AdaBoost using RBF (radial basis function) SVM base learners. In their approach, the gamma parameter of the RBF kernel was adjusted based on the training error for each base SVM classifier. They also extended their algorithm by considering the diversity of each base SVM. Experimental results were compared with a boosted neural network and a single SVM. To construct an ensemble with a large or even infinite number of base learners, Lin and Li (2005) formulated two novel kernels based on the infinite ensemble learning, which could output an infinite and non-sparse ensemble.

Although the literature presents profound insights for SVM ensemble theory and application, SVM ensembles have not been studied thoroughly like decision tree ensembles (Banfield et al., 2007; Bauer and Kohavi, 1999) or neural network ensembles (Schwenk and Bengio, 2000; Opitz and Maclin, 1999). Additionally, SVM ensembles have not been examined against a large number of data sets. On the other hand, different applications of SVM ensembles have been reported, e.g. bacterial transcription start sites prediction (Gordon et al., 2006), text-independent speaker recognition (Lei et al., 2006), fault diagnosis of roller bearings (Hu et al., 2007), land cover (Chan et al., 2001), membership authentication (Pang et al., 2003) and cardiovascular disease level prediction (Eom, et al., 2007). From this context, this paper examines SVM ensembles using a variety of ensemble constructing techniques against 20 UCI data sets. For each SVM ensemble, different kernel functions and different numbers of base learners are considered to investigate their effect upon classification performance. The results are validated against an industry case study of gear defect detection.

The remainder of the paper is organized as follows: Section 2 describes the theoretical background of support vector machines; Section 3 explains the four ensemble constructing techniques (i.e., bagging, resampling AdaBoost, resampling Arc-x4 and a modified AdaBoost) in detail; Section 4 presents the results from the tests against the 20 data sets from the UCI repository and an industry case study of gear defect detection; and Section 5 offers some concluding thoughts and the future of SVM ensembles.

## 2. Support Vector Machines

SVMs initially dealt with two-class problems. Based on the structural risk minimization (SRM) approach, support vector machines are used to construct an optimal separating hyperplane with high classification accuracy. A simple introduction of SVMs is presented here. Readers are referred to Burges (1998) and Cristianini and Shawe-Taylor (2000) for further details.

Consider a data set $\{(\mathbf{x}_i, y_i)\}$, $i = 1, 2, \ldots, N$, $N$ is the total number of samples, $y_i = \{1, -1\}$. $\mathbf{x}_i \in R^p \subset R$, i.e., $\mathbf{x}_i$ is a $p$ dimension real vector. For the linear classification, the corresponding constraint optimization model using the soft-margin method[†] is as follows:

$$\text{Minimize} \qquad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{1}$$

$$\text{Subject to} \qquad \begin{cases} y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \quad i = 1, 2, \ldots, N \\ \xi_i \geq 0 \quad i = 1, 2, \ldots, N \end{cases} \tag{2}$$

where $\xi_i$ are slack variables, measuring the degree of misclassification of the sample $\mathbf{x}_i$. $C$ is the error penalty, penalizing the non-zero $\xi_i$. The bias $b$ is a scalar, representing

[†] The basic soft-margin method SVM is employed in this work. Other SVM methods (e.g., LS-SVM, total margin based SVM, scaled SVM and fuzzy SVM) are not considered.

the bias of the hyperplane. $\mathbf{w}$ is the weight vector, defining a direction perpendicular to the hyperplane (as shown in Fig.1). The optimization problem becomes a trade-off between the margin maximization and training errors minimization.

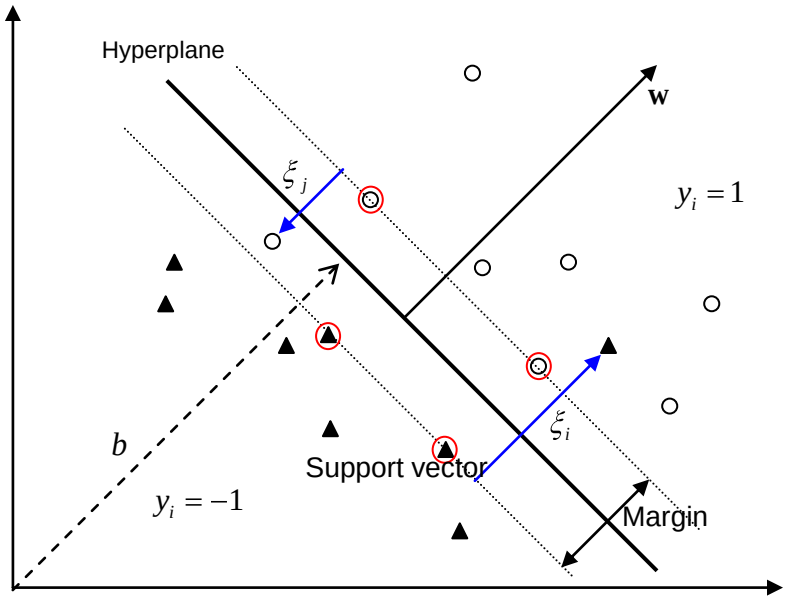


Fig.1. A geometric interpretation of the classification of SVM for non-separable data set with two classes

In particular, if the data are perfect linearly separable, then $\xi_i = 0$, and the separating hyperplane that creates the maximum distance between the plane and the nearest data (i.e., the maximum margin equals $\|\mathbf{w}\|^{-2}$) is the optimal separating hyperplane.

In general, the above model is a classical convex optimization problem (quadratic programming (QP) optimization problem). The calculation can be simplified by converting the problem into the equivalent Lagrangian dual problem:

$$\text{Minimize} \quad L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i (\mathbf{w} \cdot \mathbf{x}_i + b) + \sum_{i=1}^{N} \alpha_i \tag{3}$$

The solution of Eq. (3) can be solved by using partial derivatives of $L$ with respect to $\mathbf{w}$ and the derivation of $L$ with respect to $b$ such that the following saddle point equations can be obtained:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0 \tag{4}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{5}$$

Substituting Eq. (4) and (5) into Eq.(3), the dual quadratic optimization problem can be deduced as Eq.(6), which is to be maximized with respect to $\boldsymbol{\alpha}$, subject to Eq.(4) and (5):

$$\text{Maximize} \quad L(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=0}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{6}$$

$$\text{Subject to} \quad \begin{cases} 0 \le \alpha_i \le C \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{7}$$

According to the Karush Kuhn-Tucker (KTT) "complementarity" condition (Burges, 1998), the solution of the above dual optimization problem must satisfy the Eq.(8). This implies that

$$\alpha_i [y_i (\mathbf{w}\mathbf{x}_i + b) - 1] = 0 \tag{8}$$

for any given $i$, there will be either $\alpha_i^* = 0$ or $y_i(\mathbf{w}\mathbf{x}_i + b) = 1$ (i.e., $\alpha_i^* \neq 0$). The training data vector $\mathbf{x}_i$ corresponding to $\alpha_i^* \neq 0$ are called the support vector (SV) (as data point with red circle shown in Fig. 1). Based on the SVs, the optimal separating hyperplane can be represented as

$$f(\mathbf{x}, \alpha_i^*, b^*) = \sum_{i \in SV}^{N} \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}) + b^* \tag{9}$$

Based on the SVs, in the future testing, the decision for testing data vector $\mathbf{z}$ is as follows:

$$h(\mathbf{z}, \alpha_i^*, b^*) = \text{sgn}(\sum_{i \in SV}^{N} \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{z}) + b^*) \tag{10}$$

The model mentioned above is only for linear classification with two-class labels. To solve non-linear classification tasks, a mapping function $\Phi$ is usually employed to map the training samples from the input space into a higher-dimensional feature space. This allows the SVM to fit the maximum-margin hyperplane in the transformed feature space. In this case, the final decision function in dual form is formally similar with Eq. (10), except that every dot product in Eq. (10) is replaced by a non-linear mapping function as shown in Eq. (11.a). Using the "kernel trick" (Vapnik, 1997), a kernel function $K(\mathbf{x}_i, \mathbf{z})$ is used to substitute the dot product of mapping function $\Phi$, as shown in Eq. (11.b).

$$h(\mathbf{z}, \alpha_i^*, b^*) = \text{sgn}(\sum_{i \in SV}^{N} \alpha_i^* y_i (\Phi^T(\mathbf{x}_i) \cdot \Phi(\mathbf{z})) + b^*) \tag{11.a}$$

$$\overset{\text{kernel trick}}{=} \text{sgn}(\sum_{i \in SV}^{N} \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{z}) + b^*) \tag{11.b}$$

Any function that satisfies Mercer's theorem (Cristianini and Shawe-Taylor, 2000) can be used as a kernel function. This allows classification to be carried out in the feature space without knowing the explicit form of the mapping.

Some typical SVM kernels include linear function ($K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \cdot \mathbf{y}$), Gaussian radial basis function (RBF) ($K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|(\mathbf{x} - \mathbf{y}\|)$ where $\gamma > 0$ is related to the kernel width), polynomial function with degree $d$ and $\gamma > 0$ ($K(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} \cdot \mathbf{y}) + \gamma)^d$). Relatively speaking, the data vector with the largest norm in the training set will overwhelm all others in linear function, and even more so in polynomial function. Gaussian RBF kernel is independent of the position of the data as it only utilizes the distances between vectors. However, to obtain an optimized separating hyperplane, it is difficult to conclude that a RBF kernel outperforms linear and polynomial kernels for every data set. Therefore, all three functions are tested in this work.

To extend a basic SVM to solve multi-class classification problem with $l$ classes, one-against-one (OAO), one-against-all (OAA) and direct acyclic graph (DAG) are three

popular methods. Hsu and Lin (2002) conducted a comprehensive comparison of these three multi-class SVM classification methods, and they suggested that the one-against-one method is most suited for practical use. Therefore, in this work, one-against-one SVMs are employed as the base classifiers using the LIBSVM software (Chang and Lin, 2001).

## 3. Ensemble Constructing Techniques‡

This section describes the ensemble constructing techniques used in this work. All of the techniques combine SVM base classifiers to form different SVM ensemble classifiers.

### 3.1. Bagging

Bagging (Breiman, 1996), short for bootstrap aggregating, is a meta-algorithm to improve classification and regression models in terms of stability and classification accuracy. Although bagging is usually applied to decision tree classifiers, it can be used with any type of model.

The idea of bagging is simple and appealing: the ensemble is made of classifiers built on a bootstrap sample of the training set (Kuncheva, 2004). A bootstrap sample is generated by uniformly sampling $N'$ instances from the training set with $N$ samples with replacement ($N' \le N$). $T$ bootstrap samples $S_t$ ($t = 1, 2, \ldots, T$) are generated and the base learner SVM is trained and built from each bootstrap. A final classifier is built whose output is the class predicted most often by its sub-classifiers (e.g. majority voting), with ties broken arbitrarily (Bauer and Kohavi, 1999).

The algorithm of bagging used in this work is shown in Fig.2, where $[h_t(\mathbf{z}) = y] = \begin{cases} 1 \text{ if } h_t(\mathbf{z}) = y \\ 0 \text{ if } h_t(\mathbf{z}) \neq y \end{cases}$. The corresponding resampling subroutine is shown in Fig.3.

**Input:**

- A training set $TR = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in R^p \subset R$, $y_i \in Y = \{l_1, l_2, \ldots l_k\}$ represents class label;
- One-against-one SVM;
- Integer $T$ specifying number of iterations (i.e., the maximum number of base learners);
- Integer $N'$ ($N' \le N$) specifying number of bootstrap samples.

**Training phase:**

For $t = 1, 2, \ldots, T$

- Take a bootstrap sample $S_t$ with sample number $N'$ from the training set $TR$ using the ***resampling subroutine*** (set $w_i^{(t)} = 1/N$ for each iteration);
- Train SVM with $S_t$ and receive the hypothesis (classifier) $h_t$
- Add $h_t$ to the ensemble, $E$

**Output:** Majority voting, for a testing set $\mathbf{z}$ with class label $y \subseteq Y = \{l_1, l_2, \ldots l_k\}$,

$$h_f(\mathbf{z}) = \arg\max \sum_{t=1}^{N} [h_t(\mathbf{z}) = y]$$

Fig. 2. The bagging algorithm

‡ The code of our SVM ensembles is available upon request.

***Resampling Subroutine***

**Input:** weight vector $w_i^{(t)}$, training set $TR = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$

**Resampling Process:**

1. Set data index set $\mathbf{ITR}_i^{(t)} = \varnothing$
2. Normalize $w_i^{(t)} = w_i^{(t)} / \sum_{i=1}^{N} w_i^{(t)}$, and compute the cumulative sum vector of $w_i^{(t)}$, $\mathbf{C}_i$ ( $i = 1, 2, \ldots, N$ )
3. Generate uniformly distributed random $R_i$ ( $i = 1, 2, \ldots, N$ )
4. For $i = 1, 2, \ldots, N$
   - Find maximum value *Max* in $\mathbf{C}_i$ which is less than $R_i$, its index in $\mathbf{C}_i$ is $j$ ( $j = 1, 2, \ldots, N$ )
   - If *Max* is empty, $\mathbf{ITR}_i^{(t)} = 1$, else $\mathbf{ITR}_i^{(t)} = j + 1$

**Output:** $TR_t = TR_i \mid i = \mathbf{ITR}_i^{(t)}$

Fig.3 The resampling subroutine

**Input:** a training set $TR = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in R^p \subset R$, $y_i \in Y = \{l_1, l_2, \ldots l_k\}$ represents class label; one-against-one SVM; Integer $T$ specifying number of iterations (or the maximum number of base learners);

**Initialize:** the weight vector over $TR$ as $w_i^{(1)} = 1/N$ ( $i = 1, 2, \ldots, N$ ), $t$=1

**Training phase:**

While ( $t \le T$ )

1. Call ***Resampling subroutine***, select dataset from $TR$ with replacement to compose a new training set $TR_t = \{x_i^{(t)}, y_i^{(t)}\}_{i=1}^{N}$ for current ensemble classifier
2. Train SVM with $TR_t$, get back a hypothesis $h_t : \mathbf{X} \to \mathbf{Y}$
3. Compute the prediction error of $h_t$ on the original training set $TR$ as $\varepsilon_t = \sum_{i=1}^{N} w_i^{(t)} \quad h_t(\mathbf{x}_i) \neq y_i$
4. If $\varepsilon_t > 0.5$,

   $t = t + 1$,reset the weight vector as $w_i^{(t)} = 1/N$ ( $i = 1, 2, \ldots, N$ ) and goto step 1 (maximum 20 times, otherwise abort the loop);

   Elseif ( $0 < \varepsilon_t \le 0.5$ )

   set $\beta_t = \frac{1}{2} \ln(\frac{1-\varepsilon_t}{\varepsilon_t})$ ; Update weight vector $w_i^{(t+1)} = \frac{w_i^t}{Z_t} \beta_t^{1-[h_t(\mathbf{x}_i) \neq y_i]}$, where is a normalization constant chosen so that $w_i^{(t+1)}$ becomes a proper distribution function, $t = t + 1$

   Elseif $\varepsilon_t = 0$

   set $\beta_t = \ln(\frac{1}{10^{-10}})$ and $t = t + 1$,reset the weight vector as $w_i^{(t)} = 1/N$ ( $i = 1, 2, \ldots, N$ )

**Output:** weighted majority voting, for a testing set $\mathbf{z}$ with class label $y \subseteq Y = \{l_1, l_2, \ldots l_k\}$,

$$h_f(\mathbf{z}) = \arg\max \sum_{t=1}^{N} \beta_t [h_t(\mathbf{z}) = y]$$

Fig. 4. The AdaBoost M1 used in this work

For a given bootstrap sample, an instance in the training set has probability $1-(1-1/N')^{N'}$ of being selected at least once in the $N'$ times instances are randomly selected from the training set. For $N'=N$ and with a large enough $N$, this is about 63.2%, which means that each bootstrap sample contains only about 63.2% unique instances from the training set. This perturbation causes different classifiers to be built, which have different certain diversities.

## 3.2. Boosting

The general idea of boosting is to develop the classifier ensemble incrementally, adding one classifier at a time. The training set used for each member of the ensemble is chosen based on the performance of the earlier classifier(s) in the ensemble. In boosting, examples that are incorrectly predicted by previous classifiers are chosen more often than examples that were correctly predicted. Therefore, future learners will focus more on the examples that previous learners misclassified.

There are many boosting algorithms. In this work, three boosting algorithms are investigated to construct SVM ensemble. They are AdaBoost M1, Arc-x4 and a modified AdaBoost algorithm proposed by Zhang et al. (2007).

### 3.2.1 AdaBoost M1

AdaBoost, short for Adaptive Boosting, was formulated by Freund and Schapire (1997). It can be used in conjunction with many other learning algorithms to improve their performance. There are two approaches implemented in AdaBoost: with reweighting and with resampling. In resampling, the fixed training sample size and training examples are resampled according to a probability distribution used in each iteration. In reweighting, all training examples with weights assigned to each example are used in each iteration to train the base classifier and this technique is only useful when the weak learner can handle weighted examples (Zhang, et. al., 2007). The resampling-based AdaBoost M1 is used in this work and its algorithm is shown in Fig. 4.

### 3.2.2 Arc-x4

The Arc-x4 algorithm was proposed by Breiman (1998) to investigate whether the success of AdaBoost rests in its adaptive resampling scheme or from the final weighted combination (Kuncheva, 2004). The difference between AdaBoost and Arc-x4 is two-fold. First, the weight for a sample at step $t$ is calculated as the proportion of times $m_i\,(i=1,2,\ldots,N)$ the sample has been misclassified by the previous $t-1$ classifiers built so far. The proportion of $m_i$ has been fixed to the constant power 4. Second, the final decision is made by majority voting rather than weighted majority voting in Adaboost M1. The algorithm is shown in Fig. 5.

### 3.2.3 A Modified AdaBoost

Considering AdaBoost is quite susceptible to noise, Zhang et al. (2007) proposed a modified boosting algorithm by introducing two extra parameters. One is the sample ratio $f$ which is used to increase the overall randomness and to reduce the computational cost of the algorithm (when $f<1$), i.e., in the step 1 of Fig. 4, the number of resample examples is $fN$ rather than $N$. Another introduced parameter is $\lambda$, an annealing parameter introduced into the re-weighting process for updating probabilities assigned to training examples in each iteration to improve accuracy, i.e., $w_i^{(t+1)}=\frac{w_i^t}{Z_t}\beta_t^{(1-[h_t(\mathbf{x}_i)\neq y_i])/\lambda}$, to make the decrement (increment) of probabilities for accurately (inaccurately) predicted examples to be smaller than that in AdaBoost. Besides the modifications of these two steps, the algorithm is similar to that of AdaBoost M1 shown in Fig. 4.

**Input:** the same as in Fig.4.

**Initialize:** the weight vector over $TR$ as $w_i^{(t)} = 1/N$ ($i = 1, 2, \ldots, N$)

For $t = 1, 2, \ldots, T$

1. Call ***Resampling subroutine***, select dataset from $TR$ with replacement to compose a new training set $TR_t = \{x_i^{(t)}, y_i^{(t)}\}_{i=1}^{N}$ for current ensemble classifier
2. Train SVM with $TR_t$, get back a hypothesis $h_t : \mathbf{X} \to \mathbf{Y}$
3. Get probability distribution for selecting sample $i$ to be part of next training set

$$w_i^{(t+1)} = \frac{1 + m_i^4}{\sum_{i=1}^{N} (1 + m_i^4)}$$

**Output:** Majority voting, for a testing set $\mathbf{z}$ with class label $y \subseteq Y = \{l_1, l_2, \ldots l_k\}$,

$$h_f(\mathbf{z}) = \arg\max \sum_{t=1}^{N} [h_t(\mathbf{z}) = y]$$

Fig.5. The Arc-x4 algorithm

# 4. Numerical Experiments

## 4.1 Classification of UCI Real-world Data Sets

To compare and evaluate the performance of different SVM ensembles, 20 real-world data sets from the UCI repository were investigated as benchmarks. Table 1 gives the characteristics of these data sets. They are varied in characteristics with different numbers of classes, attributes and samples. Note that, for Breast cancer-Wisconsin, 16 samples with missing data are deleted; for Statlog (German Credit Data), the data format with 24 numerical features are used.

Before using SVMs or ensembled SVMs, parameters were first scaled between 0 and 1 using the equation $(x(i, j) - \min(x(:, j)))/(\max(x(:, j)) - \min(x(:, j)))$, where $x(i, j)$ represents the $j$ th feature in the $i$ th sample and $x(:, j)$ represents the $j$ th feature set for all samples. Table 1 also lists the scheme of the training and testing for the data sets. 10-fold cross validation was used for most data sets, and other data sets were trained and tested according to the holdout suggestion by UCI. Since the base learner was a standard soft-margin SVM without the capability of dealing with imbalanced data sets, no data set with very imbalanced samples was chosen in this work.

Three kernel functions (Gaussian RBF, polynomial and linear) were tested. For each data set, $f = 0.8$, $\lambda = 4$ were set for the modified AdaBoost according to the guidelines of Zhang et al. (2007). For each data set, an ensemble of different classifiers was trained and tested ten times and the average accuracy was reported. For cross-validation training and testing, the same folds were performed for each method under various kernel functions. For each kernel function, the same SVM parameters (as shown in Table 2) were used without any parameter optimization, as the objective was to compare the performance amongst SVM ensembles and single SVM, ceteris paribus.

The average accuracy and average standard deviation of the testing set over all 20 data sets for ensembles incorporating from 5 to 50 base SVM learners with different kernel functions are shown in Fig. 6-11. As an example, the accuracy and standard deviation (in the parentheses) of a testing set with 10 base SVM classifiers is shown in Table 3-5.

Table 1. Summary of data sets used in this paper

| Data set | Instances | Attributes | Classes | Training/Testing size | Class Distribution |
|---|---|---|---|---|---|
| **Breast cancer-Wisconsin (Origin)** | 683 | 9 | 2 | 10-fold CV | 444 for one class, 239 for second class |
| **Statlog (Australian Credit Approval)** | 690 | 14 | 2 | 10-fold CV | 383 for class 1 and 307 for class 2 |
| **Statlog (German Credit Data)** | 1000 | 24 | 2 | 10-fold CV | 700 for class 1 and 300 for class 2 |
| **Pima Indians diabetes** | 768 | 8 | 2 | 10-fold CV | 500 for class 0 and 268 for class 1 |
| **Glass Identification** | 214 | 9 | 6 | 10-fold CV | 70 for class 1; 76 for class 2; 17 for class 3; 13 for class 5; 9 for class 6 and 29 for class 7 |
| **Statlog (Heart)** | 270 | 13 | 2 | 10-fold CV | 150 for class 1 and 120 for class 2 |
| **Iris** | 150 | 4 | 3 | 10-fold CV | 50 instances for each class |
| **Statlog (Vehicle Silhouettes)** | 846 | 18 | 4 | 10-fold CV | 199 for class 1; 217 for class 2; 218 for class 3 and 212 for class 4 |
| **Connectionist Bench (Sonar)** | 208 | 60 | 2 | 10-fold CV | 97 for class 1 and 111 for class 2 |
| **Ionosphere** | 351 | 34 | 2 | 10-fold CV | 225 for good class and 126 for bad class |
| **Wine** | 178 | 13 | 3 | 10-fold CV | 59 for class1, 71 for class 2, 48 for class 3 |
| **Soybean (Small)** | 47 | 35 | 4 | 5-fold CV | 10 for class 1,2,3; 17 for class 4 |
| **Vowel Recognition** | 528 | 10 | 11 | 10-fold CV | 48 for each class |
| **Balance Scale** | 576 | 4 | 2 | 10-fold CV | 288 for each class |
| **Teaching Assistant Evaluation** | 151 | 5 | 3 | 10-fold CV | 3 roughly equal sized classes |
| **Image Segmentation** | 2310 | 19 | 7 | 210 for training, 2,100 for testing | 30 per class for training, 300 per class for testing, according to UCI |
| **Statlog (Landsat Satellite)** | 6435 | 36 | 6 | 4435 for training, 2000 for testing, | 1072 (461) for class 1, 479 (224) for 2, 961 (397) for 3, 415 (211) for 4, 470 (237) for 5, 1038 (470) for 7 |
| **Waveform-40** | 5,000 | 40 | 3 | 4,000 for training, 1,000 for testing | 33% for each of 3 classes |
| **Letter Recognition** | 20,000 | 16 | 26 | First 16,000 for training, the remaining 4,000 for testing | Roughly equal for each class |
| **Optical Recognition of Handwritten Digits** | 5,620 | 64 | 10 | 3,823 for training, 1,797 for testing | Roughly equal for each class of training and testing |

Table 2. The parameter settings of experiments

| | Parameters of SVM | Number of Classifiers $T$ |
|---|---|---|
| **Set 1** | RBF kernel function $K(\mathbf{x},\mathbf{y}) = \exp(-\gamma \|(\mathbf{x}-\mathbf{y}\|)$, $C=100$, $\gamma=2$ | $T=\{5,10,15,$ $20,25,30,$ $35,40,45,50\}$ |
| **Set 2** | Linear function $K(\mathbf{x},\mathbf{y}) = \mathbf{x}\cdot\mathbf{y}$, $C=100$ | |
| **Set 3** | Polynomial function $K(\mathbf{x},\mathbf{y}) = (\gamma(\mathbf{x}\cdot\mathbf{y})+cof0)^d$, $C=100$, $\gamma=1$, $d=2$, $cof0=1$ | |

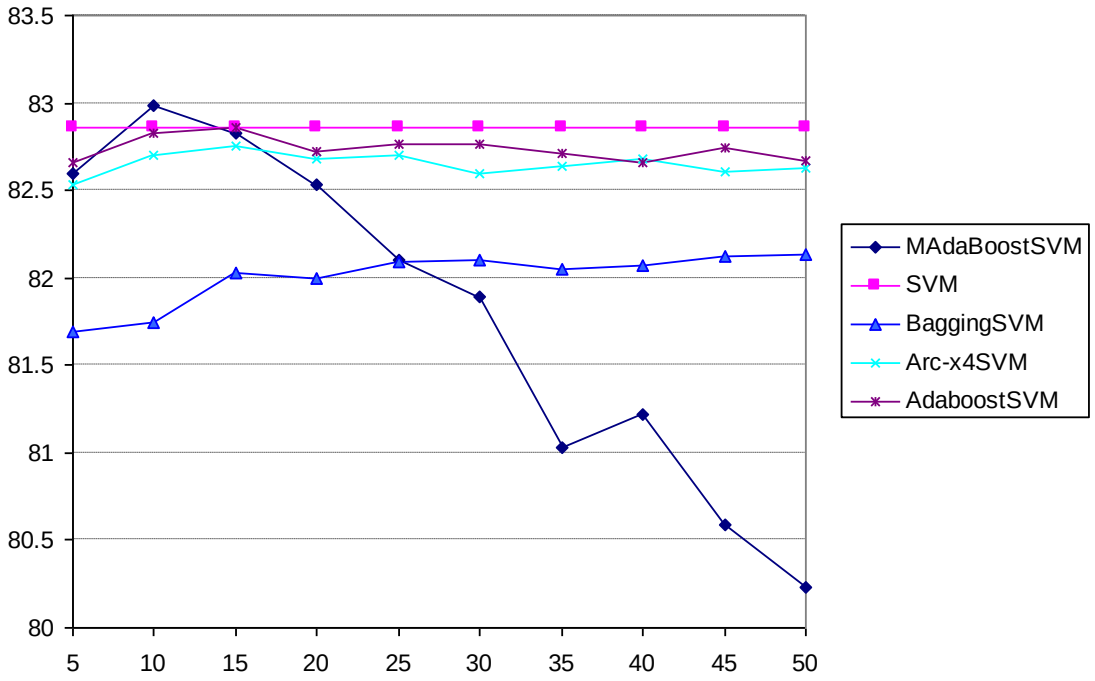


Fig. 6. Average accuracy of RBF SVM ensembles over 20 data sets

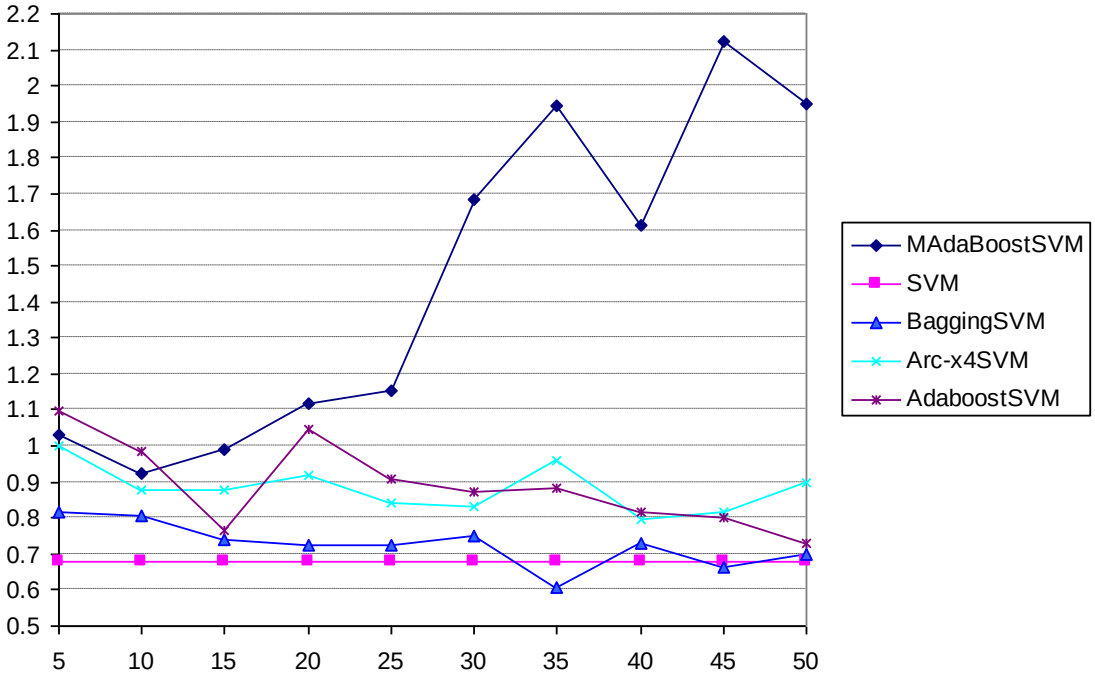

Fig. 7. Average standard deviation of RBF SVM ensembles over 20 data sets

Table 3. Classification accuracy of test sets using ten RBF kernel SVM classifiers

| Data set | MAdaBoost SVM | Single SVM | Bagging SVM | Arc-x4 SVM | AdaBoost SVM |
|---|---|---|---|---|---|
| **Breast cancer-Wisconsin (Origin)** | 95.54 (0.495) | 95.26 (0.415) | **96.57** (0.230) | 95.35 (0.440) | 95.21 (0.515) |
| **Statlog (Australian Credit Approval)** | 81.20 (0.788) | 79.86 (0.588) | **85.35** (0.351) | 78.94 (0.969) | 79.16 (0.629) |
| **Statlog (German Credit Data)** | 70.85 (0.552) | 70.72 (0.585) | **76.41** (0.415) | 70.81 (0.507) | 70.85 (0.806) |
| **Pima Indians diabetes** | 74.05 (0.982) | 74.29 (0.816) | **77.21** (0.327) | 71.43 (1.151) | 71.95 (0.787) |
| **Glass Identification** | **71.24** (1.785) | 70.48 (1.250) | 64.14 (2.590) | 69.52 (1.770) | 70.04 (1.670) |
| **Statlog (Heart)** | 78.30 (1.161) | 79 (1.210) | **83.48** (0.785) | 78.56 (1.289) | 78.41 (1.929) |
| **Iris** | 95.47 (0.878) | 95.4 (0.378) | **96.2** (0.945) | 94.4 (0.900) | 94.53 (0.820) |
| **Statlog (Vehicle Silhouettes)** | **82.77** (0.502) | 82.36 (0.682) | 80.02 (0.577) | 82.52 (0.718) | 82.54 (1.054) |
| **Connectionist Bench (Sonar)** | 82.25 (1.399) | 81.85 (1.473) | 75.85 (1.313) | 81.8 (2.002) | **82.55** (2.466) |
| **Ionosphere** | 94.51 (0.568) | **95** (0.410) | 88.94 (1.035) | 94.89 (0.342) | 94.6 (0.392) |
| **Wine** | 94.53 (1.733) | 97.88 (0.568) | 96.71 (0.632) | 97.94 (0.747) | **98.01** (0.623) |
| **Soybean (Small)** | 36.67 (1.889) | 36.67 (1.889) | **38.89** (2.147) | 36.67 (1.889) | 37.11 (3.482) |
| **Vowel Recognition** | 98.19 (1.115) | **98.96** (0.226) | 83.94 (0.786) | 98.71 (0.273) | 98.69 (0.311) |
| **Balance Scale** | **98.12** (0.573) | 97.32 (0.446) | 93.68 (0.370) | 98.09 (0.374) | 97.49 (0.543) |
| **Teaching Assistant Evaluation** | 54.87 (1.913) | **56.07** (2.361) | 52.2 (2.201) | 53.8 (2.515) | 54.87 (1.751) |
| **Image Segmentation** | 93.86 (0.454) | 93.63 (0.105) | 92.28 (0.538) | **93.86** (0.346) | 93.61 (0.309) |
| **Statlog (Landsat Satellite)** | **88.05** (0.457) | 86.23 (0.068) | 85.73 (0.270) | 87.72 (0.289) | 86.9 (0.289) |
| **Waveform-40** | 84.95 (0.344) | 83.7 (0) | **86.05** (0.276) | 84.47 (0.279) | 84.36 (0.398) |
| **Letter Recognition** | 97.21 (0.077) | 97.14 (0.063) | 84.60 (0.177) | **97.23** (0.117) | 97.10 (0.168) |
| **Optical Recognition of Handwritten Digits** | 87.11 (0.809) | 85.31 (0) | **96.60** (0.192) | 87.27 (0.683) | 88.49 (0.759) |

Table 4. Classification accuracy of test sets using ten linear kernel SVM classifiers

| Data set | MAdaBoost SVM | Single SVM | Bagging SVM | Arc-x4 SVM | AdaBoost SVM |
|---|---|---|---|---|---|
| **Breast cancer-Wisconsin (Origin)** | 96.72 (0.184) | 96.69 (0.233) | **96.78** (0.224) | 95.99 (0.250) | 96.65 (0.217) |
| **Statlog (Australian Credit Approval)** | **86.07** (0.480) | 85.25 (0.296) | 85.23 (0.285) | 83.78 (1.313) | 85.68 (0.757) |
| **Statlog (German Credit Data)** | 76.26 (0.448) | **76.59** (0.328) | 76.58 (0.399) | 73.36 (0.628) | 76.34 (0.517) |
| **Pima Indians diabetes** | 76.96 (0.418) | 77.17 (0.272) | **77.24** (0.334) | 72.92 (0.986) | 76.86 (0.639) |
| **Glass Identification** | 65 (1.827) | 64.90 (1.810) | **65.33** (1.502) | 62.95 (2.447) | 64.24 (2.075) |
| **Statlog (Heart)** | 83.19 (1.494) | **83.37** (0.790) | 83.19 (0.841) | 79.22 (1.380) | 82.11 (0.891) |
| **Iris** | 96.6 (0.492) | 96.27 (0.783) | **96.4** (0.717) | 95.2 (0.984) | 95.53 (1.045) |
| **Statlog (Vehicle Silhouettes)** | 80.44 (0.527) | 80.27 (0.514) | 80.57 (0.601) | 78.99 (0.990) | **81.13** (0.774) |
| **Connectionist Bench (Sonar)** | 75.5 (1.732) | 73.65 (2.186) | **75.55** (1.212) | 75.3 (1.670) | 74.85 (1.842) |
| **Ionosphere** | **88.83** (1.146) | 87.6 (1.168) | 88.31 (1.281) | 87.17 (1.460) | 87.34 (1.278) |
| **Wine** | 93.06 (1.408) | 96.47 (0.679) | 96.82 (0.496) | **97** (0.896) | 96.71 (1.007) |
| **Soybean (Small)** | 91.78 (4.691) | **99.56** (1.405) | 99.56 (1.405) | 99.56 (1.405) | 99.56 (1.405) |
| **Vowel Recognition** | 86.27 (1.276) | 82.42 (1.050) | 83.96 (0.885) | **89.88** (1.162) | 89.08 (0.718) |
| **Balance Scale** | 94.07 (0.420) | 94.51 (0.438) | 93.84 (0.425) | **95.37** (0.407) | 95.23 (0.601) |
| **Teaching Assistant Evaluation** | 53.73 (1.698) | **54.27** (1.265) | 52.13 (1.958) | 47.6 (2.884) | 54 (1.176) |
| **Image Segmentation** | 91.40 (0.382) | **92.77** (0.173) | 92.05 (0.654) | 91.2 (0.851) | 91.29 (0.570) |
| **Statlog (Landsat Satellite)** | 85.21 (0.399) | 85.47 (0.034) | **85.87** (0.236) | 83.82 (0.530) | 84.53 (0.366) |
| **Waveform-40** | 85.83 (0.337) | **86.26** (0.052) | 85.83 (0.298) | 85.32 (0.621) | 85.5 (0.583) |
| **Letter Recognition** | 84.32 (0.177) | 83.99 (0.117) | **84.56** (0.242) | 82.14 (0.446) | 83.91 (0.123) |
| **Optical Recognition of Handwritten Digits** | 96.64 (0.121) | 96.25 (0.105) | **96.67** (0.140) | 96.49 (0.132) | 96.49 (0.168) |

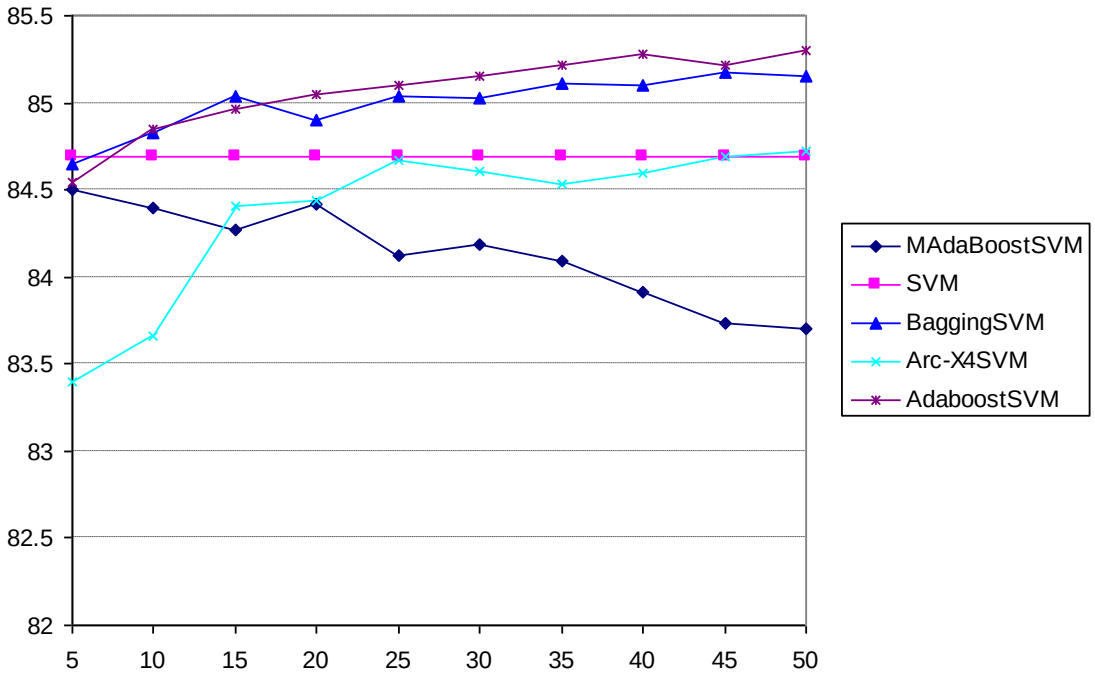


Fig.8. Average accuracy of linear SVM ensembles over 20 data sets

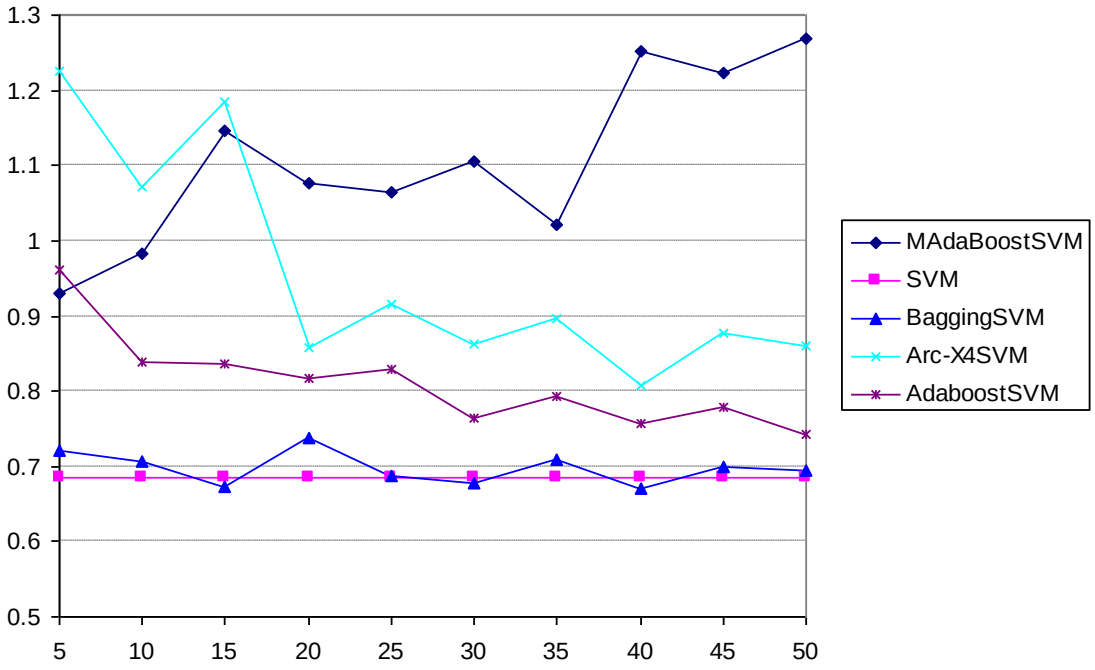


Fig. 9. Average standard deviation of linear SVM ensembles over 20 data sets

Table 5. Classification accuracy of test sets using ten polynomial kernel SVM classifiers

| Data set | MAdaBoost SVM | Single SVM | Bagging SVM | Arc-x4 SVM | AdaBoost SVM |
|---|---|---|---|---|---|
| **Breast cancer-Wisconsin (Origin)** | **95.31** (0.407) | 94.12 (0.410) | 94.85 (0.455) | 94 (0.541) | 94.37(0.444) |
| **Statlog (Australian Credit Approval)** | 84.70 (0.767) | 84.55 (0.691) | **85.09** (0.892) | 81.71(0.751) | 82.54 (0.891) |
| **Statlog (German Credit Data)** | 70.34 (0.942) | 70.55 (0.937) | **71.28** (0.666) | 69.81(0.772) | 68.95 (0.836) |
| **Pima Indians diabetes** | **76.42** (0.789) | 75.55 (0.861) | 76.18 (0.799) | 72.82 (1.372) | 75.83 (0.645) |
| **Glass Identification** | 69.62 (1.398) | 69.29 (1.172) | 69.57 (1.239) | 70.33 (2.323) | **71.67**(1.960) |
| **Statlog (Heart)** | 77.70(1.829) | 75.41(1.174) | **78.48** (1.312) | 76.26 (1.465) | 76.89 (0.785) |
| **Iris** | 96 (0.544) | **96.07** (0.734) | 95.6 (0.953) | 94.73 (0.378) | 95 (0.786) |
| **Statlog (Vehicle Silhouettes)** | 85.14 (0.400) | 84.70 (0.776) | **85.15** (0.597) | 83.76 (0.851) | 84.93 (0.792) |
| **Connectionist Bench (Sonar)** | 85.15 (1.547) | **85.6** (1.792) | 84.9 (2.092) | 85.35 (1.248) | 84.95 (0.896) |
| **Ionosphere** | 87.97(0.779) | 87.46 (0.767) | **88.11** (0.635) | 87.43 (0.571) | 87.94 (0.795) |
| **Wine** | 93.53 (1.493) | 97.18 (0.992) | 96.88 (0.879) | 97.18 (0.823) | **97.24** (1.111) |
| **Soybean (Small)** | 93.11 (4.499) | 100 (0) | 100 (0) | 100 (0) | 100 (0) |
| **Vowel Recognition** | 96.27(0.623) | 96.15 (0.790) | 95.90 (0.720) | **96.37** (0.699) | 96.35 (0.505) |
| **Balance Scale** | 99.37 (0.3) | 100 (0) | 100 (0) | 100 (0) | 100 (0) |
| **Teaching Assistant Evaluation** | 55.93 (1.762) | 56.73 (1.647) | **56.8** (1.880) | 50.67 (2.244) | 55.47 (2.218) |
| **Image Segmentation** | 92.19 (0.401) | 92.09 (0.221) | **92.52** (0.458) | 91.96 (0.556) | 91.46 (0.341) |
| **Statlog (Landsat Satellite)** | 83.85 (0.949) | 84.75 (0.136) | **85.94** (0.766) | 81.51 (0.643) | 81.43 (0.580) |
| **Waveform-40** | **84.11** (0.351) | 82.24 (0.052) | 83.92 (0.496) | 83.08 (0.480) | 82.96 (0.259) |
| **Letter Recognition** | 94.84 (0.146) | 95.06 (0.134) | 94.75 (0.132) | 95.13 (0.112) | **95.18** (0.167) |
| **Optical Recognition of Handwritten Digits** | 97.35 (0.139) | 97.25 (0.198) | **97.42** (0.166) | 97.35 (0.182) | 96.68 (1.653) |

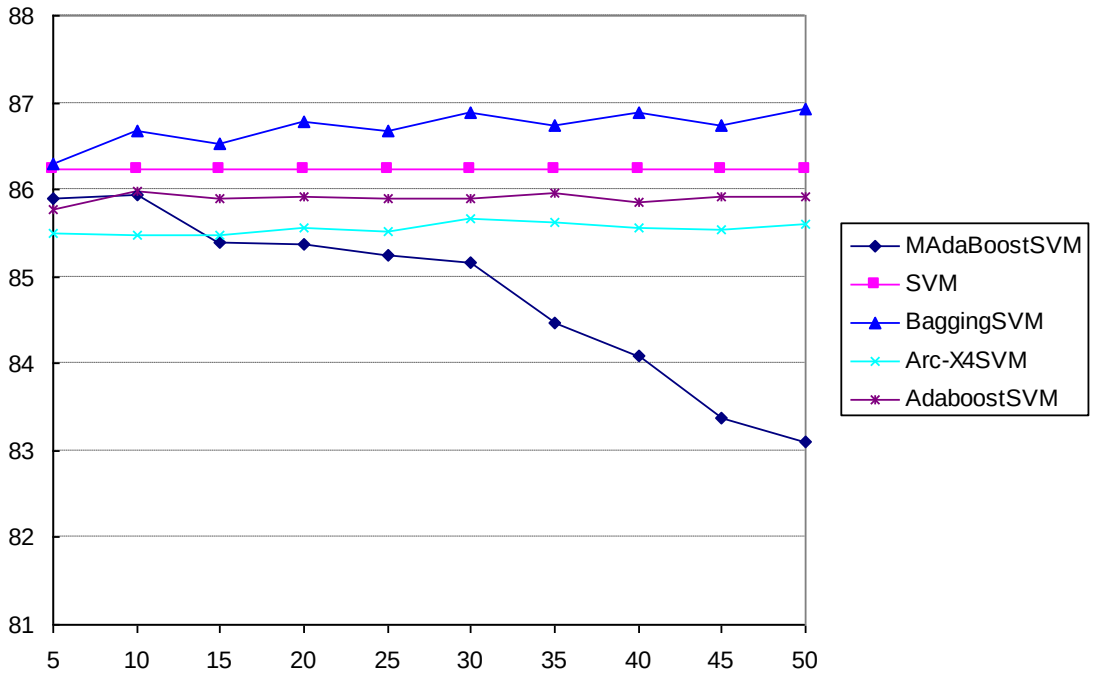


Fig.10. Average accuracy of polynomial SVM ensembles over 20 data set

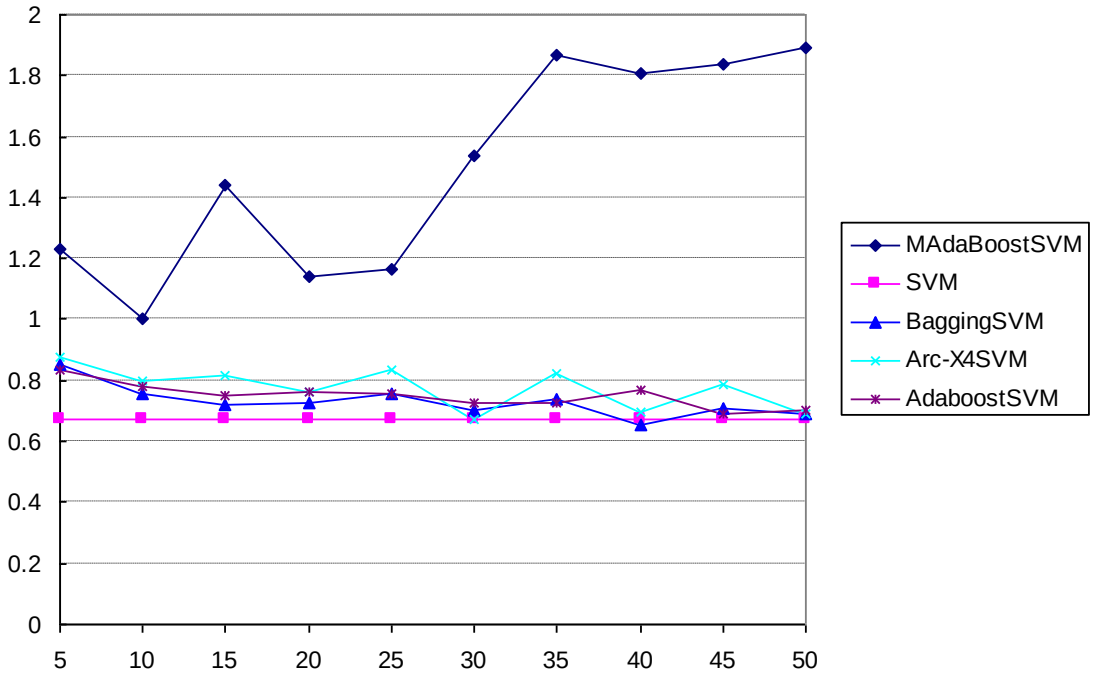


Fig. 11. Average standard deviation of polynomial SVM ensembles over 20 data sets

## 4.2 Discussion

The results demonstrate that SVM ensembles are not always better than a single SVM classifier. However, the average overall accuracies of BaggingSVM were better than those of other ensembles and a single SVM in the cases of linear and polynomial kernels. In three cases, the average standard deviations of BaggingSVM were all better than those of other ensembles. The performance of MAdaBoostSVM decreases when the number of classifiers increases beyond ten. The performance of AdaboostSVM was better than those of Arc-x4SVM in terms of average overall accuracy for all three cases.

In the case of the RBF kernel function, the average overall accuracy of BaggingSVM was worse than a single SVM, Arc-x4SVM and AdaboostSVM; the results of the BaggingSVM for different data sets were not as stable as the linear and polynomial kernels. In this situation, a single SVM had relatively greater classification accuracy, although MAdaBoostSVM performed the best with a ten SVM ensemble.

As shown in Tables 3 to 5, BaggingSVM performed the best in 9, 8, 9 out of 20 data sets for RBF, linear and polynomial kernel functions, respectively. As a general technique for ensembled SVMs, bagging with a polynomial kernel function appears to provide the best performance and generalization. Therefore in the next subsection, the performance of BaggingSVM with polynomial kernel function is compared to a single SVM and AdaBoostSVM for the practical case of gear defect detection.

## 4.3 An Industrial Case of Gear Detection

A case study involving an automotive gearbox was used to test the SVM ensembles. Resonant inspection (RI) is a technique used to measure the structural response of a metal gear and evaluate it against the statistical variation from a control set of good parts (Chen et al., 2007). A crack in a gear will change the stiffness of its neighboring regions and dampen vibration propagation, and changes in either of these attributes reflected in changes to the structure's resonant frequencies and their corresponding amplitudes. However, methods for frequency shifts and amplitude damping may ignore or miss many deviation patterns from measured data. Therefore, in this paper, we will test the accuracy of the SVM ensembles based on the resonant frequencies and their corresponding amplitudes.

Table.6. The raw data of the gear detection problem

| No. | Var $1_{rf}$ | Var $1_a$ | Var $2_{rf}$ | Var $2_a$ | Var $3_{rf}$ | Var $3_a$ | … | Var $14_{rf}$ | Var $14_a$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 8546.9 | 0.2327 | 15296.9 | 0.3589 | 17203.1 | 0.0998 | … | 46437.5 | 0.0023 |
| 2 | 8531.2 | 0.0786 | 15203.1 | 0.2417 | 17171.9 | 0.0939 | … | 46281.2 | 0.0022 |
| 3 | 8562.5 | 0.2087 | 15328.1 | 0.4016 | 17218.8 | 0.0623 | … | 46437.5 | 0.0034 |
| 4 | 8531.2 | 0.3013 | 15359.4 | 0.1563 | 17218.8 | 0.0743 | … | 46562.5 | 0.0009 |
| 5 | 8531.2 | 0.0336 | 15234.4 | 0.1585 | 17187.5 | 0.0839 | … | 46265.6 | 0.0025 |
| 6 | 8531.2 | 0.0702 | 15218.8 | 0.3625 | 17203.1 | 0.0453 | … | 46421.9 | 0.0011 |
| 7 | 8531.2 | 0.021 | 15218.8 | 0.2495 | 17156.2 | 0.0633 | … | 46359.4 | 0.001 |
| 8 | 8531.2 | 0.053 | 15187.5 | 0.2347 | 17171.9 | 0.0614 | … | 46375 | 0.0027 |
| 9 | 8531.2 | 0.1552 | 15375 | 0.5647 | 17203.1 | 0.0955 | … | 46203.1 | 0.0008 |
| 10 | 8562.5 | 0.0934 | 15296.9 | 0.2658 | 17203.1 | 0.054 | … | 46437.5 | 0.0007 |
| … | … | … | … | … | … | … | … | … | … |
| 6973 | 8562.5 | 0.3115 | 15343.8 | 0.2007 | 17203.1 | 0.0413 | … | 46484.4 | 0.0007 |

The structural resonant responses of 6973 gears were measured, amongst which 674 gears were known to be faulty. Table 6 shows fourteen pairs of resonant frequencies and their corresponding amplitudes, named Var $X_{rf}$ and Var $X_a$ respectively for the $x^{th}$ pair of resonant frequencies and their corresponding amplitudes. Chen et al. (2007) used an information entropy based feature selection and self-organizing map (SOM) method to solve this problem. In this paper, SVM ensembles are used albeit without feature selection, to see its potential capability in handling large feature spaces (Widodo and Yang, 2007).

As the data was imbalanced (6299 non-defect vs. 674 defect samples), a holdout training scheme was used instead of the standard n-fold cross validation. Considering the basic SVM is a standard soft-margin SVM, the same number of samples from the defect and non-defect gear data were taken to form the training data set, while the remainder was used for testing. Different combinations of the training data set were checked (as shown in the first column in Table 7. 50-50 means that 50 defect and 50 non-defect samples were selected randomly to form the training data set. For each combination, the computation was

Table 7. The $TN_{rate}$ and $TP_{rate}$ values of three classifiers for gear detection

| | | Single SVM | | BaggingSVM | | AdaBoostSVM | |
|---|---|---|---|---|---|---|---|
| | | True Negative | True Positive | True Negative | True Positive | True Negative | True Positive |
| **50-50** | Avg. | **95.27** | 98.48 | 95.19 | **98.66** | 95.13 | 98.47 |
| | Std. | 2.26 | 0.73 | 2.26 | **0.58** | **2.19** | 0.73 |
| **100-100** | Avg. | 97.31 | 98.61 | **97.59** | **98.87** | 97.41 | 98.82 |
| | Std. | 1.41 | 0.45 | 1.18 | **0.31** | **1.09** | 0.39 |
| **150-150** | Avg. | 98.09 | 99.14 | **98.13** | **99.25** | 97.98 | 99.24 |
| | Std. | 0.85 | 0.25 | **0.69** | **0.25** | 0.77 | 0.25 |
| **200-200** | Avg. | 98.32 | 99.25 | **98.49** | **99.31** | 98.36 | 99.31 |
| | Std. | 0.89 | 0.2 | **0.78** | 0.21 | 0.78 | **0.19** |
| **250-250** | Avg. | 98.49 | 99.36 | **98.64** | 99.36 | 98.51 | **99.38** |
| | Std. | 0.54 | 0.27 | **0.51** | 0.22 | 0.54 | **0.21** |
| **300-300** | Avg. | 98.81 | 99.41 | **98.89** | **99.43** | 98.82 | 99.43 |
| | Std. | 0.51 | 0.18 | **0.5** | **0.16** | 0.54 | 0.21 |
| **350-350** | Avg. | **99.01** | 99.43 | 98.97 | **99.5** | 98.93 | 99.43 |
| | Std. | 0.55 | 0.13 | 0.48 | 0.12 | **0.48** | **0.1** |
| **400-400** | Avg. | 98.74 | 99.45 | **98.86** | 99.45 | 98.77 | **99.46** |
| | Std. | 0.63 | 0.15 | **0.48** | **0.13** | 0.58 | 0.14 |

repeated ten times, and the average True Negative Rate ($TN_{rate}$) and True Positive Rate ($TP_{rate}$) were recorded. For each combination, the training data set was then selected randomly 20 times (i.e. for each algorithm, 200 computations were executed) to determine statistically significant results. The average and standard deviation value of $TN_{rate}$ and $TP_{rate}$ for single SVM, BaggingSVM and AdaboostSVM are listed in the Table 7. Here, a polynomial kernel function was used with $C=100$, $\gamma=1$, $d=2$, cof0=1.

From the results in Table 7, it was found that a single SVM, BaggingSVM and AdaBoostSVM exhibit similar performance. However, the results obtained by the BaggingSVM ensemble were slightly better in average than those obtained by a single SVM, though the improvement was only 0.25% and 0.15% for $TP_{rate}$ and $TN_{rate}$ respectively. However, for more than 6,000 testing data samples, even a 0.1% improvement is still significant in theory. Moreover, the standard deviation of BaggingSVM was lower than single SVM. However, in practice, a single SVM may be enough to deal with this case.

The G-mean (geometric mean) (Kubat. et al., 1998) can be used as a performance accuracy measure that combines the True Positive Rate and True Negative Rate for this two-class classification problem. The G-mean is defined as:

$$\text{G-mean} = \sqrt{TP_{rate} \times TN_{rate}}$$

The G-means for the results are shown in Table 8, and show that BaggingSVM outperformed the other two techniques for this case.

## 5. Conclusions

Table 8. G-mean value of three classifiers for the gear detection

| | Single SVM | BaggingSVM | AdaBoostSVM |
|---|---|---|---|
| **50-50** | 96.88 | **96.91** | 96.79 |
| **100-100** | 98.16 | **98.33** | 98.21 |
| **150-150** | 98.67 | **98.69** | 98.61 |
| **200-200** | 98.81 | **98.90** | 98.83 |
| **250-250** | 98.92 | **99.00** | 98.94 |
| **300-300** | 99.13 | **99.16** | 99.12 |
| **350-350** | 99.23 | **99.23** | 99.18 |
| **400-400** | 99.09 | **99.15** | 99.11 |

An extensive experimental evaluation of several ensemble methods with SVM classifier was presented in this paper. Bagging, AdaBoost, Arc-X4, and a modified AdaBoost were compared against a standard soft-margin SVM classifier using the experimental results of 20 data sets in UCI repository and an industrial case of gear defect detection. The results demonstrated that although SVM ensembles are not always better than single SVM for every data set, the SVM ensemble methods on average resulted in a better classification accuracy than a single SVM. Moreover, among SVM ensembles, bagging is considered the most appropriate ensemble technique for most problems for its relatively better performance and higher generality.

For practical applications, the selection between a single SVM and SVM ensemble results in a tradeoff between the incremental performance gains and the processing time costs. There is a risk for SVM ensembles – their additional computational time does not guarantee performance improvement and can sometimes be detrimental to accuracy. Therefore, in future SVM ensemble research, a greater emphasis should be placed on selecting appropriate SVM ensembles based on the scenario characteristics to ensure a greater performance improvement.

## Acknowledgements

This research was supported by the NSF Industry/University Cooperative Research Center (I/UCRC) for Intelligent Maintenance Systems (IMS) at the University of Cincinnati, University of Michigan and University of Missouri-Rolla. This work was also supported by the National Science Foundation of China under Grant #60574054, the Programme of Introducing Talents of Discipline to Universities (B06012) and the Program for New Century Excellent Talents in University of China (NCET 2006). The authors also want to express their appreciation of Dr. Haixia Wang's assistance.

### References

Banfield, R.E., Hall, L.O., Bowyer, K.W. and Kegelmeyer, W.P. A comparison of decision tree ensemble creation techniques. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2007, 29(1): 173-180.

Bauer E. and Kohavi R. An empirical comparison of voting classification algorithms: bagging, boosting and variants. Machine Learning, 1999, 36: 105-139.

Breiman L. Random forests. Machine Learning, 2001, 45:5-32.

Breiman, Arcing Classifiers, The Annals of Statistics, 1998, 26(3): 801-849.

Breiman, L. Bagging predictors. Machine Learning, 1996, 24, 123-140.

Burges, C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 1998, 2: 121-167.

Chan J., Huang C. and DeFries R. Enhanced algorithm performance for land cover

classification from remotely sensed data using bagging and boosting. IEEE Transactions on Geoscience and Remote Sensing, 2001, 39(3): 693-695.

Chang C.C. and Lin C.J. LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Chen Y., Wang H.X. and Lee, J. A New Method for Feature Selection and Gear Defect Detection. In ASME International Conference on Manufacturing Science & Engineering (MSEC), 2007, Atlanta, GA, US.

Cristianini N and Shawe-Taylor J. A Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, Cambridge, UK, 2000.

Eom, J-H., Kim S-C and Zhang B-T. AptaCDSS-E: A classifier ensemble-based clinical decision support system for cardiovascular disease level prediction. Expert System with Application, 2007, in press.

Freund Y. Boosting a weak learning algorithm by majority. Information and Computation, 1995, 121(2): 256-285.

Freund Y. and Schapire R. A desicion-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 1997, 55(1): 119-139.

Freund Y. and Schapire R. A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence, 1999, 14(5): 771-780.

Gordon J.J., Towsey M.W., Hogan J.M., Mathews S.A. and Timms P. Improved prediction of bacterial transcription start sites. Bioinformatics, 2006, 22(2): 142-148.

Hsu C-W and Lin C-J. A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks, 2002, 13(2): 415-425.

Hu Q., He Z., Zhang, Z. and Zi Y. Fault diagnosis of rotating machinery based on improved wavelet package transform and SVMs ensemble. Mechanical Systems and Signal Processing, 2007, 21: 688-705.

Kim Y-C., Pang S., Je H-M., Kim D. and Bang S-Y. Constructing support vector machine ensemble. Pattern Recognition, 2003, 36: 2757-2767.

Kubat M., Holte R., Matwin S. Machine learning for the detection of oil spills in satellite radar images. Machine Learning, 1998, 30: 195-215.

Kuncheva, L.I. Combining pattern classifiers: methods and algorithms. John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.

Lei Z., Yang Y. and Wu Z. Ensemble of support vector machine for text-independent speaker recognition. International Journal of Computer Science and Network Security, 2006, 6(5A): 163-167.

Li X., Wang L. and Sung E. A study of AdaBoost with SVM based weak learners. Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on. 2005, vol. 1: 196- 201.

Lin, H-T and Li L. Novel distance-based SVM kernels for infinite ensemble learning. In Proceedingsof the 12th International Conference on Neural Information Processing, 2005, p.761-766.

Opitz, D. and Maclin R. Popular ensemble methods: an empirical study. Journal of Artificial Intelligence Research, 1999, 11: 169-198.

Pang S, Kim D and Sung Y. Membership authentication in the dynamic group by face classification using SVM ensemble. Pattern Recognition Letter, 2003, 24: 215-225.

Schwenk H. and Bengio Y. Boosting Neural Network. Neural Computation, 2000, 12: 1869-1887.

UCI Machine Learning Repository. http://archive.ics.uci.edu/beta/datasets.html

Valentini, G. and Dietterich T. Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research,

2004, 5: 725-775.

Valentini G. An experimental bias-variance analysis of SVM ensembles based on resampling techniques. IEEE Transactions on System, Man and Cybernetics-Part B: Cybernetics, 2005, 35(6): 1252-1271.

Vapnik, V. The Nature of Statistical Learning Theory, 2nd Edition, Springer-Verlag, New York, 1997.

Webb, G.I. and Zheng Z. Multistrategy ensemble learning: reducing error by combining ensemble learning technique. IEEE Transaction on Knowledge and Data Engineering, 2004, 16(8):980-991.

Webb G.I. MultiBoosting: a technique for combing boosting and wagging, machine learning, 2000, 40(2): 159-196.

Wezel M. and Potharst R. Improved customer choice predictions using ensemble methods. European Journal of Operational Research, 2007, 181: 436-452.

Widodo, A., Yang B-S and Han T. Combination of independent component analysis and support vector machines for intelligent faults diagnosis of induction motors. Expert Systems with Application, 2007, 32: 299-312.

Widodo A. and Yang B-S. Support vector machine in machine condition monitoring and fault diagnosis. Mechanical Systems and Signal Processing, 2007, in press.

Zhang C.X., Zhang J.S. and Zhang G.Y. An efficient modified boosting method for solving classification problems. Journal of Computational and Applied Mathematics, 2007, online.